Graphical and Algebraic Representatives of Conditional Independence Models

Jiří Vomlel and Milan Studený

Institute of Information Theory and Automation,
Academy of Sciences of the Czech Republic,
Pod vodárenskou věží 4,
182 08 Prague 8, Czech Republic

Abstract. The topic of this chapter is conditional independence models. We review mathematical objects that are used to generate conditional independence models in the area of probabilistic reasoning. More specifically, we mention undirected graphs, acyclic directed graphs, chain graphs, and an alternative algebraic approach that uses certain integer-valued vectors, named imsets. We compare the expressive power of these objects and discuss the problem of their uniqueness.

In learning Bayesian networks one meets the problem of non-unique graphical description of the respective statistical model. One way to avoid this problem is to use special chain graphs, named essential graphs. An alternative algebraic approach uses certain imsets, named standard imsets, instead. We present algorithms that make it possible to transform graphical representatives into algebraic ones and conversely. The algorithms were implemented in the R language.

1 Conditional Independence Models

The main motivation for this chapter is conditional independence models (CI models). The classical concept of independence of two variables has an interpretation of their mutual irrelevance, which means that knowing more about the state of the first variable does not have any impact on our knowledge of the state of the second variable. Similarly, the concept of conditional independence of two variables given a third variable means, that if we know the state of the third variable then knowing more about the state of the first variable does not have any impact on our knowledge of the state of the second variable. We will illustrate this concept using an example taken from Jensen (2001).

Example 1 (CI model). Assume three variables:

- person's length of hair, denoted by h,
- person's stature, denoted by s, and
- person's gender, denoted by g.

We can describe relations between these three variables as follows:

- Seeing the *length of hair* of a person will tell us more about his/her *gender* and conversely. It means, the value of g is dependent on the value of h.

- Knowing more about the *gender* will focus our belief on his/her *stature*. It means that s is dependent on g and (through g) also on h.
- Nevertheless, if we know the *gender* of a person then *length of hair* of that person gives us no extra clue on his/her *stature*, that is h is independent of s given g.

Thus, we have indicated one (conditional) independence relation. Note that all the observations are implicitly understood as symmetric claims. That is, for example, the dependence between g and h is the same as the dependence between h and g.

The conditional independence relations of the above-mentioned type can formally be specified as conditional independence statements over a non-empty finite set of variables N.

Definition 1 (Disjoint triplet over N). Let $A, B, C \subseteq N$ be pairwise disjoint subsets of a set of variables N. This *disjoint triplet over* N will be denoted by $\langle A, B \mid C\rangle$. The symbol $\mathcal{T}(N)$ will be used to denote the class of all possible disjoint triplets over N.

Definition 2 (CI statement). Let $\langle A, B \mid C\rangle$ be a disjoint triplet over N. Then the statement "A is conditionally independent of B given C" is a *CI statement* (over N), written as $A \perp\!\!\!\perp B \mid C$. The negation of this statement, the respective conditional dependence statement, will be denoted by $A \not\perp\!\!\!\perp B \mid C$.

If any of the sets A,B, or C will be a one-element set we will omit the curly brackets and write a to denote $\{a\}$.

Example 2 (CI statement). In Example 1 we have indicated only one CI statement, $h \perp\!\!\!\perp s \mid g$. On the other hand, we have indicated two dependence statements, namely $g \not\perp\!\!\!\perp h$ and $s \not\perp\!\!\!\perp g$. These are viewed as conditional dependence statements, namely $g \not\perp\!\!\!\perp h \mid \emptyset$ and $s \not\perp\!\!\!\perp g \mid \emptyset$.

Using CI statements we can create a model of a real domain/system that describes conditional independence relations between all variables in the modeled domain.

Definition 3 (CI model). A *CI model* is a set of CI statements.

Example 3 (CI model). The situation in Example 1 can be modelled by a CI model containing just one non-trivial CI statement $h \perp\!\!\!\perp s \mid g$. However, one should also include trivial CI statements, which have the form $A \perp\!\!\!\perp \emptyset \mid C$, where $A, C \subseteq \{s, h, g\}$ are disjoint. They correspond to an intuitively evident statement that there cannot be any dependency on an empty set of variables. It is implicitly understood that the disjoint triplets that are not present in the list of CI statements are interpreted as conditional dependence statements.

One way to describe an independence model of a real system is to provide the list of all valid CI statements. However, for large domains of interest this list might be terribly long. This means, humans would hardly be able to to create, maintain, or verify the correctness of a list of this kind.

The list of CI statements can be shortened if we take into consideration certain properties of probabilistic CI models. The CI models that satisfy five basic axioms listed below are called *semi-graphoids* (Pearl, 1988). These axioms can be used as rules that generate from a certain set of CI statements called the *base* all valid CI statements in the modeled system. Thus, instead of storing the whole list of CI statements we just keep the CI statements in the base.

Definition 4 (Semi-Graphoid Axioms). For each collection $A, B, C, D \subseteq N$ of pairwise disjoint sets the following axioms are assumed:

$$\begin{aligned} A \perp\!\!\!\perp \emptyset \mid C, & \\ A \perp\!\!\!\perp B \mid C &\implies B \perp\!\!\!\perp A \mid C, \\ A \perp\!\!\!\perp B \cup D \mid C &\implies A \perp\!\!\!\perp B \mid C, \\ A \perp\!\!\!\perp B \cup D \mid C &\implies A \perp\!\!\!\perp B \mid C \cup D, \\ A \perp\!\!\!\perp B \mid C \cup D \ \wedge \ A \perp\!\!\!\perp D \mid C &\implies A \perp\!\!\!\perp B \cup D \mid C. \end{aligned}$$

Even if lists of CI statements are shortened so that they only contain CI statements in a base they still might be hardly understandable by humans. Therefore various auxiliary mathematical objects were proposed that can be used to generate CI models.

In Section 2 we review some of these mathematical objects that are traditionally used in the area of probabilistic reasoning. More specifically, we discuss *undirected graphs*, *acyclic directed graphs*, *chain graphs*, and an alternative algebraic approach that uses certain integer-valued vectors, named *imsets*. We compare their expressive power and discuss the problem of their uniqueness. The discussion is a starting point for introducing imsets since they meet two requirements:

- so-called standard imsets can represent each CI model generated by a discrete probability distribution, and
- imsets from a special class of standard imsets are unique representatives of CI models generated by acyclic directed graphs.

These properties play an important role in learning CI models.

In Section 3 we introduce two representatives of an equivalence class of acyclic directed graphs – *essential graphs* and *standard imsets*. The core of this chapter is the transition between these two representatives of Bayesian network models. Therefore, in Section 4, we explain in detail why an algorithm for this transition is desired in the area of learning Bayesian networks. Then, in Section 5, certain graphical characteristics of chain graphs are introduced that play a crucial role in the reconstruction of the essential graph on

basis of the standard imset. In Section 6 we formulate lemmas on which the reconstruction algorithm is based and the algorithm is given in Section 7. In Section 8 we discuss the relation to hierarchical junction trees, which provide a third method for representing a Bayesian network model. We implemented all mentioned algorithms in the R language developed by R Development Core Team (2004). In Conclusions we discuss future perspectives.

2 Objects generating conditional independence models

The main motivation for using mathematical objects to generate CI models is that they often have much more compact form when compared with a list of CI statements. They are also better readable by humans. By a *CI object* over N we will understand a mathematical object defined over a finite set (of variables) N that can be used to generate a CI model.

In this section we review the most popular classes of CI objects: discrete probability distributions, undirected graphs, acyclic directed graphs, chain graphs, and imsets. We will describe how they generate CI models and compare their expressive power with respect to the collection of all CI models generated by discrete probability distributions. All CI models discussed in this section are semi-graphoids.

2.1 Probability distributions

Every discrete probability distribution (PD) defined over variables from N can be used to induce a CI model over N. In Definition 5 the concept of conditional independence for discrete PDs is recalled.

Definition 5 (CI in PDs). Let P be a discrete PD over N. Given any $A \subseteq N$, let v_A denote a configuration of values of variables from A and for $B \subseteq N \setminus A$ let $P(v_A \mid v_B)$ denote the conditional probability for $A = v_A$ given $B = v_B$. Given $\langle A, B \mid C \rangle \in \mathcal{T}(N)$, the CI statement $A \perp\!\!\!\perp B \mid C$ is induced by probability distribution P over N if for all v_A, v_B, v_C such that $P(v_C) > 0$

$$P(v_A, v_B \mid v_C) = P(v_A \mid v_C) \cdot P(v_B \mid v_C) \ . \tag{1}$$

Example 4. Assume that h, s, g are variables from Example 1. For simplicity, further assume that they are all discrete and binary:

- h taking states *short* and *long*,
- s taking states *more than 164 cm* and *less than 164 cm*, and
- g taking states *male* and *female*.

Let $A = \{h\}$, $B = \{s\}$, and $C = \{g\}$. If a probability distribution $P(A, B, C)$ satisfies equation (1) for all respective configuration of values (v_A, v_B, v_C) from

$$\{\textit{short, long}\} \times \{\textit{more than 164 cm, less than 164 cm}\} \times \{\textit{male, female}\}$$

then $A \perp\!\!\!\perp B \mid C$ (which is $h \perp\!\!\!\perp s \mid g$) is a CI statement induced by P.

2.2 Undirected graphs

Probabilistic graphical models modelled by undirected graphs are also known as *Markov networks* (Pearl, 1988).

Definition 6 (UG). An *undirected graph* (UG) over N is a pair $(N, \mathcal{E})$, where N is a set of nodes and $\mathcal{E}$ a set of undirected edges, i.e., a set of unordered pairs $a - b$ where $a, b \in N$ and $a \neq b$.

Definition 7 (Undirected path). An *undirected path* between nodes a and b in an UG $G = (N, \mathcal{E})$ is a sequence $a \equiv c_1, \ldots, c_n \equiv b$, $n \geq 2$ of nodes such that $c_1, \ldots, c_n$ are distinct and $c_i - c_{i+1} \in \mathcal{E}$ for $i = 1, \ldots, n-1$.

Definition 8 (Separation criterion for UGs). $A \perp\!\!\!\perp B \mid C$ is represented in an UG G if every path in G between a node in A and a node in B contains a node from C.

Example 5. The UG in Figure 1 represents the CI statement $h \perp\!\!\!\perp s \mid g$.

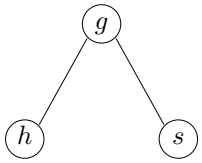

Fig. 1. The UG generating $h \perp\!\!\!\perp s \mid g$

Definition 9 (Clique). A set of nodes $K \subseteq N$ of an UG $G = (N, \mathcal{E})$ is complete if $a - b \in \mathcal{E}$ for all $a, b \in K, a \neq b$. A maximal complete set of G with respect to set inclusion is called a *clique*.

An important concept is that of a *decomposable* (undirected) graph. There are several equivalent definitions of a decomposable graph (see § 2.1.2 of Lauritzen (1996)), one of them is the following.

Definition 10 (Decomposable graph). An undirected graph $G = (N, \mathcal{E})$ is *decomposable* if its cliques can be ordered into a sequence $K_1, \ldots, K_m$, $m \geq 1$ satisfying the *running intersection property* (cf. Proposition 2.17(ii) in (Lauritzen, 1996)):

$$\forall\, i \geq 2 \;\; \exists\, k < i \quad S_i \equiv K_i \cap (\bigcup_{j<i} K_j) \subseteq K_k\,. \tag{2}$$

It is a well-known fact that the collection of sets S_i, $2 \leq i \leq m$ does not depend on the choice of an ordering satisfying (2) – see Lemma 7.2 in Studený (2005b). We will call these sets *separators* of the graph. Moreover, the multiplicity $\nu(S)$ of a separator S, that is, the number of indices i for which $S = S_i$ also does not depend on the choice of an ordering satisfying (2). Note that the definition implies that the class of cliques is disjoint with the class of separators.

Example 6. The UG in Figure 1 has two cliques $C_1 = \{h, g\}$ and $C_2 = \{g, s\}$. It is decomposable and it has one separator $S = \{g\}$.

2.3 Acyclic directed graphs

Acyclic directed graphs are a very popular class of CI objects in the area of probabilistic reasoning. They constitute the structural part of probabilistic models known as *Bayesian network models* (Pearl, 1988).

Definition 11 (Directed graph). A *directed graph* is a pair $(N, \mathcal{F})$, where N is the set of nodes and $\mathcal{F}$ is a set of directed edges, i.e., a set of ordered pairs $a \to b$ where $a, b \in N$ and $a \neq b$ such that if $a \to b \in \mathcal{F}$ then $b \to a \notin \mathcal{F}$.

Definition 12 (Directed path). A *directed path* between nodes a and b in a directed graph $G = (N, \mathcal{F})$ is a sequence $a \equiv c_1, \ldots, c_n \equiv b, n \geq 2$ such that $c_1, \ldots, c_n$ are distinct and for $i = 1, \ldots, n-1$: $c_i \to c_{i+1} \in \mathcal{F}$.

Definition 13 (Directed cycle). A *directed cycle* in a graph $G = (N, \mathcal{F})$ is a sequence $c_1, \ldots, c_n, c_{n+1} \equiv c_1, n \geq 3$ of nodes in G such that $c_1, \ldots, c_n$ are distinct and for $i = 1, \ldots, n-1$: $c_i \to c_{i+1} \in \mathcal{F}$.

Definition 14 (DAG). An *acyclic directed graph* (DAG) is a directed graph that has no directed cycle.

Remark 1. Researchers in the area of artificial intelligence became accustomed to the abbreviation DAG. It is based on the phrase *directed acyclic graph*, which is, however, imprecise from the grammatical point of view.

Definition 15 (Underlying graph of a DAG). *Underlying graph* of a DAG $G = (N, \mathcal{F})$ is undirected graph $G' = (N, \mathcal{E})$, where

$$\mathcal{E} = \{a - b : a \to b \in \mathcal{F} \vee a \to b \in \mathcal{F}\} \quad .$$

Definition 16 (Set of parents in a DAG). Let $G = (N, \mathcal{F})$ be a DAG. The set of *parents* of a node $b \in N$ is the set $\{a \in N : a \to b \in \mathcal{F}\}$, denoted by $pa_G(b)$.

To introduce so-called moralization criterion for reading CI statements from a DAG we will use graphical concepts of *ancestral set* and *moral graph*.

Definition 17 (Ancestral set in a DAG). Let $G = (N, \mathcal{F})$ be a DAG. Node a is an *ancestor* of node b in G if there exists a directed path from a to b. Given $B \subseteq N$, the set of ancestors of nodes in a set of nodes B is called the (least) *ancestral set* for B. It will be denoted $an_G(B)$.

Definition 18 (Induced subgraph for a DAG). Assume a DAG $G = (N, \mathcal{F})$ and $\emptyset \neq M \subseteq N$. The *induced subgraph* of G for M is graph $G_M = (M, \mathcal{F}_M)$, where $\mathcal{F}_M = \{a \to b \in \mathcal{F} : a \in M, b \in M\}$.

Note that every induced subgraph of a DAG is again a DAG.

Definition 19 (Moralization for DAGs). The *moral graph* of a DAG $G = (N, \mathcal{F})$ is graph G^{mor}, which is an undirected graph $(N, \mathcal{E})$, where

$$\mathcal{E} = \{a - b : (a \to b \in \mathcal{F}) \vee (b \to a \in \mathcal{F}) \vee (\exists c \in N \setminus \{a, b\} : a \to c, b \to c \in \mathcal{F})\}$$

Now, we can formulate the *moralization criterion*. To check whether a CI statement is represented in a DAG, we first create an ancestral graph, moralize it, and finally check the validity of the CI statement using the separation criterion for UGs as defined in Definition 8.

Definition 20 (Moralization criterion for DAGs). $A \perp\!\!\!\perp B \mid C$ is represented in a DAG G if $A \perp\!\!\!\perp B \mid C$ is represented in the moral graph of induced graph $G_{an_G(A \cup B \cup C)}$.

Example 7. In Figure 2 three DAGs generating the CI statement $h \perp\!\!\!\perp s \mid g$ are shown. They all have the same moral graph, which is in Figure 1. Note that there are three different DAGs that generate the same CI model.

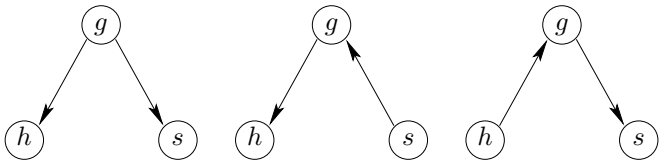

Fig. 2. Three DAGs generating $h \perp\!\!\!\perp s \mid g$

2.4 Chain graphs

Chain graphs (Lauritzen and Wermuth, 1989) are a common generalization of both UGs and DAGs. They form a special subclass of graphs that contain both directed and undirected edges, so called *hybrid graphs*.

Definition 21 (Hybrid graph). A *hybrid graph* is a triplet $(N, \mathcal{E}, \mathcal{F})$, where N is the set of nodes, $\mathcal{E}$ is a set of undirected edges, i.e., a set of unordered pairs $a - b$ where $a, b \in N$ and $a \neq b$ and $\mathcal{F}$ is a set of directed edges, i.e., a set of ordered pairs $a \to b$ where $a, b \in N$ and $a \neq b$. Moreover, it is required that if $a \to b \in \mathcal{F}$ then $b \to a \notin \mathcal{F}$ and $a - b \notin \mathcal{E}$.

Definition 22 (Semi-directed cycle). A *semi-directed cycle* in a hybrid graph H is a sequence $c_1, \ldots, c_n, c_{n+1} \equiv c_1, n \geq 3$ of nodes in G such that $c_1, \ldots, c_n$ are distinct, $c_1 \to c_2 \in \mathcal{F}$, and either $c_i \to c_{i+1} \in \mathcal{F}$ or $c_i - c_{i+1} \in \mathcal{E}$ for $i = 2, \ldots, n$.

Definition 23 (Chain graph). A *chain graph* (CG) is a hybrid graph that has no semi-directed cycle.

Example 8. In Figure 3 we give an example of a chain graph.

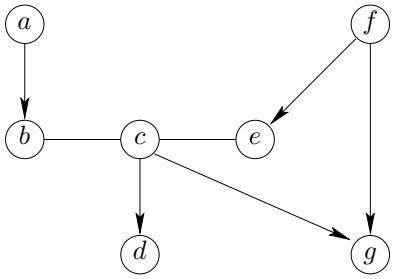

Fig. 3. A chain graph

The following definitions generalize definitions of undirected and directed paths.

Definition 24 (Path). *Path* in a hybrid graph $H = (N, \mathcal{E}, \mathcal{F})$ between nodes a and b is a sequence $a \equiv c_1, \ldots, c_n \equiv b, n > 1$ such that $c_1, \ldots, c_n$ are distinct and for $i = 1, \ldots, n-1$ one of the following three conditions holds:

(1) $c_i - c_{i+1} \in \mathcal{E}$,
(2) $c_i \to c_{i+1} \in \mathcal{F}$, or
(3) $c_{i+1} \to c_i \in \mathcal{F}$.

If for $i = 1, \ldots, n-1$ either the condition (1) or (2) holds the path is called *descending*.

Example 9. An example of a descending path in the chain graph in Figure 3 is the path $a \to b - c \to d$.

Next, we generalize the concepts of set of parents and ancestral set for CGs.

Definition 25 (Set of parents in a CG). Let $H = (N, \mathcal{E}, \mathcal{F})$ be a CG. The *set of parents* of nodes in a set B is the set $\{a \in N : a \to b \in \mathcal{F}, b \in B\}$. It will be denoted $pa_H(B)$.

Definition 26 (Ancestral set in a CG). Let $H = (N, \mathcal{E}, \mathcal{F})$ be a CG. Node a is an *ancestor* of node b in H if there exists a descending path from a to b. The set of ancestors of nodes in a set of nodes B is called the (least) *ancestral set* for B. It will be denoted $an_H(B)$.

Example 10. Let H be the chain graph from Figure 3. For example, the set of parents of $\{d\}$ is $\{c\}$, the ancestral set of $\{d\}$ is $\{a, b, c, e, f\}$. The ancestral set of $\{d, g\}$ is the set of all nodes N.

Next, we generalize special concepts from DAGs to CGs.

Definition 27 (Induced subgraph for a CG). Let $H = (N, \mathcal{E}, \mathcal{F})$ be a CG and $\emptyset \neq M \subseteq N$. The *induced subgraph* of H for M is graph $H_M = (M, \mathcal{E}_M, \mathcal{F}_M)$, where $\mathcal{E}_M = \{a - b \in \mathcal{E} : a, b \in M\}$ and $\mathcal{F}_M = \{a \to b \in \mathcal{F} : a, b \in M\}$.

Note that every induced subgraph of a CG is a CG.

Definition 28 (Components). A set of nodes $C \subseteq N$ is *connected* in a CG $H = (N, \mathcal{E}, \mathcal{F})$ if for all $a, b \in C$ there exists a path in the undirected graph $G = (N, \mathcal{E})$. Maximal connected subsets of N with respect to set inclusion are called *components* in H. The class of all components in H will be denoted $\mathcal{C}(H)$.

Remark 2. Observe that every induced subgraph of a CG H for a component $C \in \mathcal{C}(H)$ is an UG. Given a $C \in \mathcal{C}(H)$, $pa_H(C)$ is disjoint with C. Components in a CG $H = (N, \mathcal{E}, \mathcal{F})$ form a partition of N. Components in a DAG are singletons.

Example 11. The chain graph H from Figure 3 has five components. Thus, $\mathcal{C}(H) = \{\, \{a\}, \{f\}, \{b, c, e\}, \{d\}, \{g\} \,\}$.

Definition 29 (Moralization). The *moral graph* of a hybrid graph $H = (N, \mathcal{E}, \mathcal{F})$ is graph H^{mor}, which is the undirected graph $(N, \mathcal{E})$, where

$$\mathcal{E} = \{a - b : (a \to b \in \mathcal{F}) \vee (b \to a \in \mathcal{F}) \vee (a - b \in \mathcal{E}) \vee (\exists C \in \mathcal{C}(H) : a \neq b,\ \{a, b\} \subseteq pa_H(C))\}$$

Example 12. In Figure 4 we give the moral graph of the graph from Figure 3.

Now, we are ready to introduce the moralization criterion for reading CI statements represented in a CG. To check a CI statement, we first create an ancestral graph, moralize it, and finally check the CI statement using the separation criterion for UGs as defined in Definition 8.

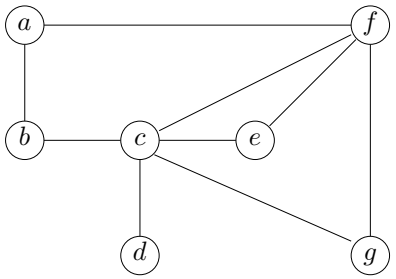

Fig. 4. The moral graph of the chain graph from Figure 3

Definition 30 (Moralization criterion for CGs).
CI statement $A \perp\!\!\!\perp B \mid C$ is represented in a CG H if $A \perp\!\!\!\perp B \mid C$ is represented in the moral graph of induced graph $H_{an_H(A \cup B \cup C)}$.

Remark 3. Note that CGs are generalization of DAGs and UGs because the moralization criterion for CGs generalizes the separation criterion for UGs and the moralization criterion for DAGs.

Example 13. In order to verify whether the CI statement $a \perp\!\!\!\perp d \mid \{b, e, g\}$ is represented in the chain graph H in Figure 3 we check this CI statement using the separation criterion in the moral graph G of the ancestral graph for the set $\{a, b, d, e, g\}$, which is apparently the undirected graph in Figure 4. Since there exist a path from a to d that does not contain a node from $\{b, e, g\}$

$$a - f - c - d$$

the CI statement is not represented in the undirected graph G and, consequently, it is not represented in the CG H.

2.5 Imsets

An *imset* (Studený, 2005b) is an algebraic object that can be used to describe a CI model.

Definition 31 (Imset). Let N be a finite set, $\mathcal{P}(N) = \{A : A \subseteq N\}$ the power set of N, and $\mathbb{Z}$ the set of all integers. An *imset* u is a function $u : \mathcal{P}(N) \mapsto \mathbb{Z}$.

Remark 4. Let $\mathbb{N}$ be the set of all natural numbers. Sometimes, function $m : \mathcal{P}(N) \mapsto \mathbb{N}$ is called multiset. The word *imset* is an abbreviation for Integer valued MultiSET.

Zero function on the power set of N will be denoted by 0. Note that every imset can be interpreted as a vector, whose components are integers indexed by subsets of N.

Example 14. Let $N = \{h, g, s\}$. An example of an imset over N is

B	$u(B)$
$\emptyset$	0
$\{h\}$	0
$\{g\}$	+1
$\{s\}$	0
$\{h, g\}$	−1
$\{g, s\}$	−1
$\{h, s\}$	0
$\{h, g, s\}$	+1

We will specify imsets using a special convention. Let us use Kronecker's symbol δ to denote an indicator function defined for $A, B \subseteq N$ as follows:

$$\delta_A(B) = \begin{cases} 1 & \text{if } A = B, \\ 0 & \text{otherwise.} \end{cases}$$

Then, given an imset u, one has

$$\forall B \subseteq N: \; u(B) = \sum_{A \subseteq N} u(A) \cdot \delta_A(B)$$

which can be abbreviated as

$$u = \sum_{A \subseteq N} c_A \cdot \delta_A$$

where $c_A = u(A) \in \mathbb{Z}$ is a the respective coefficient for every $A \subseteq N$.

Example 15. Using the convention we will write the imset from Example 14 as follows:

$$u = \delta_{\{g\}} - \delta_{\{h,g\}} - \delta_{\{g,s\}} + \delta_{\{h,g,s\}}$$

Definition 32 (Elementary imset). Let $K \subseteq N$, $a, b \in N \setminus K$, and $a \neq b$. The *elementary imset* corresponding to $\langle a, b \mid K \rangle$ is given by the formula

$$u_{\langle a,b|K\rangle} = \delta_{\{a,b\}\cup K} + \delta_K - \delta_{\{a\}\cup K} - \delta_{\{b\}\cup K} \; .$$

The symbol $\mathcal{E}(N)$ will denote the set of all elementary imsets over N.

Definition 33 (Structural imset). An imset u is *structural* iff

$$n \cdot u = \sum_{v \in \mathcal{E}(N)} k_v \cdot v \; ,$$

where $n \in \mathbb{N}$ and $v \in \mathcal{E}(N) : k_v \in \mathbb{N} \cup \{0\}$.

As an analogy to graphical criteria in the case of UGs, DAGs, and CGs, we need a criterion that specifies how imsets generate CI models. This time we have an algebraic criterion.

Definition 34. CI statement $A \perp\!\!\!\perp B \mid C$ is represented in a structural imset u over N if there exists $k \in \mathbb{N}$ such that $k \cdot u = u_{\langle A,B|C\rangle} + w$, where w is a structural imset.

2.6 Comparison of CI objects

The CI objects discussed above are often used to represent CI models induced by probability distributions. Thus, we may compare how well they can play this role. Let

- $M(P)$ be a CI model generated by a discrete probability distribution P over N,
- $\mathcal{M}$ denote the class of all CI models generated by discrete probability distributions over N ,
- $\mathcal{O}$ denote a considered class of CI objects (e.g., UGs, DAGs, CGs, structural imsets).

Definition 35 (Perfectly Markovian). A probability distribution P is *perfectly Markovian* with respect to an object $O \in \mathcal{O}$ if for every disjoint triplet $\langle A, B \mid C \rangle \in \mathcal{T}(N)$

$$A \perp\!\!\!\perp B \mid C \text{ is represented in } P \iff A \perp\!\!\!\perp B \mid C \text{ is represented in } O \ .$$

In other words, perfect Markovness means that P and O generate the same CI model. Now, we can raise three basic questions about the relation between a class $\mathcal{O}$ of CI objects and the class $\mathcal{M}$ of CI models generated by discrete probability distributions.

Definition 36 (Faithfulness). A class of CI objects $\mathcal{O}$ is *faithfull* if for every CI object $O \in \mathcal{O}$ there exists a CI model $M(P)$ from $\mathcal{M}$ such that P is perfectly Markovian with respect to O.

Definition 37 (Completeness). A class of CI objects $\mathcal{O}$ is *complete* if for every CI model $M(P)$ from $\mathcal{M}$ there exists a CI object $O \in \mathcal{O}$ such that P is perfectly Markovian with respect to O.

Definition 38 (Uniqueness). A class of CI objects $\mathcal{O}$ satisfies the *uniqueness* property if for every CI model $M(P)$ from $\mathcal{M}$ *at most one* CI object $O \in \mathcal{O}$ exists such that P is perfectly Markovian with respect to O.

Table 1 compares properties of different classes of CI objects. In contrary to graphical probabilistic models (UGs, DAGs, and CGs) the class of structural imsets is complete, that is, it can describe all CI models generated by discrete PDs.

Table 2 shows the numbers of different CI models that can be generated by different classes of CI objects. We can see that, already for N having only four elements, only a small fraction of CI models generated by discrete PDs over N can be represented by DAGs, UGs, and CGs. Structural imsets are complete, which means that they can represent all CI models from $\mathcal{M}$.

Table 1. Properties of classes of CI objects

	Faithfulness	Completeness	Uniqueness
UGs	yes	no	yes
DAGs	yes	no	no
CGs	yes	no	no
structural imsets	no	yes	no

Table 2. Comparison of the number of CI models generated by different classes of CI objects for $|N| = 3, 4, 5$. The number of CI models generated by discrete PDs is not known for $|N| \geq 5$.

$\|N\|$	UGs	DAGs	CGs	$\|\mathcal{M}\|$
3	8	11	11	22
4	64	185	200	18300
5	1024	8782	11519	?

3 Representatives of equivalence classes of DAGs

We have already mentioned that several different DAGs may generate the same CI-model, that is, DAGs do not satisfy the uniqueness property. This unpleasant fact may cause some problems in learning Bayesian networks (see Section 4) and motivates the need for a uniquely determined representative of the respective CI model. In this section we present two unique representatives of an equivalence class of DAGs that generate the same CI model. First, we define the *essential graph* and note how it can be constructed from a DAG. Second, we introduce a uniquely determined algebraic representative of an equivalence class of DAGs – the *standard imset*.

Definition 39 (Independence equivalence). Two CI-objects are called *independence equivalent* if they define the same CI-model. We will briefly say that they are equivalent.

Verma and Pearl (1991) gave a direct graphical characterization of equivalent DAGs. It uses the following concept.

Definition 40 (Immorality). An *immorality* in a DAG $G = (N, \mathcal{F})$ is an induced subgraph of G for a set $\{a, b, c\} \subseteq N$ such that $a \to c$, $b \to c \in \mathcal{F}$ and $a \leftarrow b, a \to b \notin \mathcal{F}$.

Lemma 1. *(Verma and Pearl, 1991)*
Two acyclic directed graphs are equivalent iff they have the same underlying graph and immoralities.

3.1 Essential graphs

Definition 41 (Essential graph). The *essential graph* (EG) G^* of an equivalence class $\mathcal{G}$ of DAGs over N is a hybrid graph over N defined as follows:

- $a \to b$ in G^* if $a \to b$ in G for every $G \in \mathcal{G}$,
- $a - b$ in G^* if $\exists\, G_1, G_2 \in \mathcal{G}$ such that $a \to b$ in G_1 and $a \leftarrow b$ in G_2.

We say that a hybrid graph H over N is an EG over N if there exists an equivalence class $\mathcal{G}$ of DAGs over N such that $H = G^*$.

Of course, this definition is in terms of the whole equivalence class of DAGs. Nevertheless, there exists an algorithm for getting the EG on basis of any $G \in \mathcal{G}$ – see Studený (2004). A graphical characterization of EGs was presented by Andersson et al. (1997a). To recall it we need the following notion.

Definition 42 (Flag). A *flag* in a CG $H = (N, \mathcal{E}, \mathcal{F})$ is an induced subgraph of H for a set $\{a, b, c\} \subseteq N$ such that $a \to b \in \mathcal{F}$, $b - c \in \mathcal{E}$, and $a \leftarrow c, a \to c \notin \mathcal{F}, a - c \notin \mathcal{E}$.

Example 16. The chain graph in Figure 3 has two flags, $a \to b - c$ and $f \to e - c$.

Lemma 2. *(Andersson et al., 1997a, Theorem 4.1)*
A hybrid graph $H = (N, \mathcal{E}, \mathcal{F})$ is an EG iff it is a CG without flags such that, for every component $C \in \mathcal{C}(H)$, the induced subgraph H_C is decomposable and, for every $a \to b \in \mathcal{F}$, at least one of the following conditions holds:

- $\exists c \in N : (c \to a \in \mathcal{F}) \wedge (c \to b \notin \mathcal{F})$,
- $\exists c \in N : (c \to b \in \mathcal{F}) \wedge (c - a \notin \mathcal{E}) \wedge (c \to a \notin \mathcal{F})$,
- $\exists c_1, c_2 \in N : (c_1 \to b, c_2 \to b \in \mathcal{F}) \wedge (c_1 - a, c_2 - a \in \mathcal{E}) \wedge$
  $(c_1 \to c_2, c_2 \to c_1 \notin \mathcal{F}) \wedge (c_1 - c_2 \notin \mathcal{E})$

Example 17. An example of an EG is given in Figure 5.

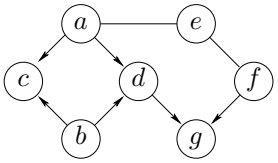

Fig. 5. Example of an EG.

Another useful result is a characterization of CGs equivalent to DAGs, which uses the following concept.

Definition 43 (Closure graph). If C is a component in a CG H then by the *closure graph* for C we will understand the moral graph of the induced subgraph for the set $D = C \cup pa_H(C)$. It will be denoted by $\bar{H}(C)$.

Lemma 3. *(Andersson et al., 1997b, Proposition 4.2)*
A chain graph H is equivalent to a DAG iff, for every component C of H, the closure graph $\bar{H}(C)$ is decomposable. In particular, the induced subgraph H_C is decomposable for any $C \in \mathcal{C}(H)$.

3.2 Standard imsets

Another uniquely determined representative of an equivalence class of DAGs is the *standard imset* (Studený, 2005b).

Definition 44 (Standard imset). Given a DAG $G = (N, \mathcal{F})$, the *standard imset* for G is given by the formula

$$u_G = \delta_N - \delta_\emptyset + \sum_{a \in N} \left\{ \delta_{pa_G(a)} - \delta_{\{a\} \cup pa_G(a)} \right\} \quad . \tag{3}$$

Note that some terms in the formula (3) can cancel each other and some terms can be merged together. The basic observation is as follows.

Lemma 4. *(Studený, 2005b, Corollary 7.1)*
Two DAGs G and H are independence equivalent iff $u_G = u_H$.

Thus, standard imsets can serve as unique representatives of the respective CI model. Another pleasant fact is that standard imsets, viewed as vectors, have many zero components. Therefore, they can effectively be kept in the memory of a computer.

Now, we give a formula for the standard imset on basis of any chain graph H over N which is equivalent to a DAG. It is based on Lemma 3. Let $\bar{\mathcal{K}}(C)$ denote the collection of cliques of $\bar{H}(C)$ and $\bar{\mathcal{S}}(C)$ the collection of separators in $\bar{H}(C)$. Further, let $\bar{\nu}_C(S)$ denote the multiplicity of a separator S in $\bar{H}(C)$.

The standard imset for H is given by the following formula:

$$u_H = \delta_N - \delta_\emptyset + \sum_{C \in \mathcal{C}(H)} \{ \delta_{pa_H(C)} - \sum_{\bar{K} \in \bar{\mathcal{K}}(C)} \delta_{\bar{K}} + \sum_{\bar{S} \in \bar{\mathcal{S}}(C)} \bar{\nu}_C(\bar{S}) \cdot \delta_{\bar{S}} \}. \tag{4}$$

The point is that the formula (4) gives the same result for equivalent chain graphs.

Lemma 5. *(Studený et al., 2005, Proposition 20)*
Let G and H are equivalent chain graphs such that there exists a DAG equivalent to them. Then $u_G = u_H$.

Of course, if $H = G$ is a DAG, then (4) gives the same result as (3). On the other hand, since the EG G^* of an equivalence class of DAGs $\mathcal{G}$ is a CG equivalent to any $G \in \mathcal{G}$, we can conclude this:

Corollary 1. *Let G be a DAG and H the EG of the respective equivalence class. Then the formula (4) gives the standard imset for G.*

Example 18. In Table 3 we give the standard imset for the EG from Figure 5.

B	$u(B)$
$\{a,b,c,d,e,f,g\}$	$+1$
$\emptyset$	$+1$
$\{a,b\}$	$+2$
$\{d,f\}$	$+1$
$\{a,b,c\}$	-1
$\{a,b,d\}$	-1
$\{d,f,g\}$	-1
$\{b\}$	-1
$\{a,e\}$	-1
$\{e,f\}$	-1
$\{e\}$	$+1$

Table 3. The standard imset for the EG from Figure 5. The values for remaining subsets of N are zero.

4 Learning Bayesian networks

Specific motivation for the transition between EGs and standard imsets is learning Bayesian networks. A *Bayesian network model* has two components:

- *structure*, determined by a DAG, whose nodes correspond to variables,
- *parameters*, namely the numbers in the collection of conditional probability tables, which correspond to the DAG.

We are interested in learning structure of a Bayesian network from data. Actually, our aim is to determine the respective *statistical model*, that is, the class of probability distributions with prescribed structure.

4.1 Quality criterion

The basic division of methods for learning Bayesian networks is as follows:

- methods based on *statistical conditional independence tests*.
- methods based on the *maximization of a quality criterion*.

In this chapter, we are interested in maximization of a quality criterion

$$\mathcal{Q} : \mathsf{DAGS}\,(N) \times \mathsf{DATA}\,(N, d) \longrightarrow \mathbb{R}$$

where $\mathsf{DAGS}\,(N)$ is the class of DAGs over N and $\mathsf{DATA}\,(N, d)$ is the collection of databases over N of the length $d \geq 1$. In this context, the graph "represents" the respective statistical model. Because the direct maximization of a quality criterion is typically infeasible, the researchers in artificial intelligence developed various *methods of local search*, see Chickering (2002).

The basic idea of these methods is to introduce the concept of *neighbourhood* for representatives of considered CI models (= graphs) and search for a local maximum of the criterion with respect to the neighbourhood structure. Typically, the change in the value of a (common reasonable) quality criterion is easy to compute. Natural neighbourhood concept from a mathematical point of view is so-called *inclusion neighbourhood*, see Kočka and Castelo (2001).

4.2 Problem of representative choice

This topic is related to the question of internal computer representation of a Bayesian network model. There are two approaches:

- to use any DAG in the respective equivalence class (which need not be unique),
- to use a suitable uniquely determined representative.

We prefer using a unique representative. This is because we believe that non-uniqueness may lead to computational inefficiencies. Another reason is that using a unique representative is more elegant from a mathematical point of view. Two possible unique representatives, namely the EG and the standard imset, were already mentioned in Section 3.

A natural question is whether one can "translate" one to the other. Our special motivation is as follows. The inclusion neighbourhood of a given Bayesian network model is already characterized in terms of the EG – see Studený (2005a). We would like to have its characterization in terms of the standard imset. Below we explain why we consider standard imsets particularly suitable for this purpose.

4.3 Algebraic approach

The basic idea of an algebraic approach (to learning Bayesian networks) is to use the standard imset as a unique representative of the respective statistical model – se § 8.4 in (Studený 2005). The advantage of this approach is that every imset can be interpreted as a vector and (reasonable) quality criteria appear to be affine (= shifted linear) functions of the standard imset.

More specifically, every criterion $\mathcal{Q}(G, D)$ depending on a DAG G and a database D, which is score-equivalent and decomposable Studený (2005b), has the form

$$\mathcal{Q}(G, D) = s_D^{\mathcal{Q}} + \sum_{S \subseteq N} t_D^{\mathcal{Q}}(S) \cdot u_G(S),$$

where $s_D^{\mathcal{Q}}$ is a constant depending on data and $[t_D^{\mathcal{Q}}(S)]_{S \subseteq N}$ is so-called *data vector* (relative to a criterion $\mathcal{Q}$).

Example 19. In the case of a well-known Bayesian information criterion (= BIC) one has the following formula for the respective data vector:

$$t_D^{\mathsf{BIC}}(S) \;=\; d \cdot H(\hat{P}_S | \prod_{i \in S} \hat{P}_i) - \frac{1}{2} \cdot \ln d \cdot \{\, |S| - 1 + \prod_{i \in S} r(i) - \sum_{i \in S} r(i) \,\},$$

where $\emptyset \neq S \subseteq N$, d is the length of the database D, $H(*|*)$ denotes the relative entropy, $\hat{P}_S$ is the marginal of the empirical measure (based on D) for S and $r(i) = |\mathsf{X}_i|$, $i \in S$ are the cardinalities of the respective individual sample spaces. Moreover, one has $t_D^{\mathsf{BIC}}(\emptyset) = 0$.

Another pleasant fact is that, in the method of local search, the *move* between two neighbouring model in the sense of inclusion neighbourhood is characterized by a simple elementary imset, which is the difference of respective standard imsets. Therefore, the move can be interpreted in terms of a CI statement $a \perp\!\!\!\perp b \,|\, C$. In particular, the respective change in the value of $\mathcal{Q}$ takes a neat form of the scalar product of two vectors:

$$\langle t_D^{\mathcal{Q}}, u_{\langle a,b|C \rangle} \rangle = \sum_{S \subseteq N} t_D^{\mathcal{Q}}(S) \cdot u_{\langle a,b|C \rangle}(S)\,.$$

5 Graphical characteristics of chain graphs

The formula (4) can be simplified for chain graphs without flags. In this section, we introduce some characteristics of these graphs that will be used in a simplified formula given in Section 6. The proofs of claims from Sections 5.1 and 5.2 can be found in Studený et al. (2005).

5.1 Initial components

Definition 45 (Initial component). A component $C \in \mathcal{C}(H)$ in a CG H such that $pa_H(C) = \emptyset$ will be called an *initial component* in H. Let us denote by $i(H)$ the number of initial components in H.

Note that $i(H) \geq 1$ and this number appears to be the same for equivalent CGs.

Example 20. The chain graph H in Figure 5 has two initial components: $\{b\}$ and $\{a, e, f\}$. Thus, $i(H) = 2$.

5.2 Core

Definition 46 (Idle set). We will say that a set B of nodes in a CG $H = (N, \mathcal{E}, \mathcal{F})$ is *idle* if the following two conditions hold:

- $\forall\, b_1, b_2 \in B,\ b_1 \neq b_2, \quad (b_1 \to b_2 \in \mathcal{F}) \vee (b_2 \to b_1 \in \mathcal{F}) \vee (b_1 - b_2 \in \mathcal{E})$
- $\forall\, a \in N \setminus B, \forall\, b \in B, \quad a \to b$ in H.

The meaning of these conditions is that no non-trivial CI statement represented in H involves variables in B. The second condition implies that $\forall\, C \in \mathcal{C}$ if $C \cap B \neq \emptyset$ then $C \subseteq B$. Therefore, every idle set is the union of some components. One can easily show that every CG H over N has a unique maximal idle set of nodes, possibly empty. This set can be shown to be the same for equivalent chain graphs.

Definition 47 (Core, core-components). The complement $N \setminus B$ of the maximal idle set B will be called the *core* of H and denoted by $core(H)$. The class of *core-components*, that is, components in H contained in the core, will be denoted by $\mathcal{C}_{core}(H)$.

Observe that if the core is non-empty then every initial component is a core-component.

5.3 Cliques and separators

If H is a CG without flags equivalent to a DAG then every its core-component C induces a decomposable graph H_C by Lemma 3. Let us denote by $\mathcal{K}(C)$ the class of its cliques, by $\mathcal{S}(C)$ the collection of its separators, and by $\nu_C(S)$ the multiplicity of $S \in \mathcal{S}(C)$ in H_C. Note that, the fact that C is connected implies that every $S \in \mathcal{S}(C)$ is a non-empty proper subset of $\mathcal{C}_{core}(H)$.

Example 21. The chain graph H in Figure 5 has an empty maximal idle set, i.e., the core is $core(H) = N$. Its core-components are $C_1 = \{a, e, f\}$, $C_2 = \{b\}$, $C_3 = \{c\}$, $C_4 = \{d\}$ and $C_5 = \{g\}$. All components except for C_1 have only one clique and no separator. The set of cliques of H_{C_1} is $\mathcal{K}(C_1) = \{\{a, e\}, \{e, f\}\}$ and the set of its separators is $\mathcal{S}(C_1) = \{\{e\}\}$.

5.4 Parent sets

Definition 48 (Parent sets). A set $P \subseteq N$ will be called a *parent set* in a CG H if it is non-empty and there exists a core-component $C \in \mathcal{C}_{core}(H)$ with $P = pa_H(C)$. The *multiplicity* $\tau(P)$ of a parent set P is the number of $C \in \mathcal{C}_{core}(H)$ with $P = pa_H(C)$. Let us denote the collection of parent sets in H by $\mathcal{P}_{core}(H)$.

Evidently, every $P \in \mathcal{P}_{core}(H)$ is a proper subset of $\mathcal{C}_{core}(H)$.

Example 22. The parent sets in the chain graph H in Figure 5 are $\{a, b\}$ and $\{d, f\}$. The multiplicities are $\tau(\{a, b\}) = 2$ and $\tau(\{d, f\}) = 1$.

6 Simplified formula for imset of essential graph

For CGs without flags the formula (4) can be simplified as follows.

Lemma 6. *(Studený and Vomlel, 2005, Lemma 5.1).*
Let H be a CG without flags equivalent to a DAG with $\mathcal{C}_{core}(H) \neq \emptyset$. Then the standard imset for H is given by

$$u_H = \delta_{core(H)} - \sum_{C \in \mathcal{C}_{core}(H)} \sum_{K \in \mathcal{K}(C)} \delta_{K \cup pa_H(C)} + \sum_{S \in \mathcal{S}(C)} \nu_C(S) \cdot \delta_{S \cup pa_H(C)} + \sum_{P \in \mathcal{P}_{core}(H)} \tau(P) \cdot \delta_P + \{i(H) - 1\} \cdot \delta_\emptyset \,.$$

The point is that, in the case of a non-trivial EG H, none of the terms in the above formula cancel each other.

Lemma 7. *(Studený and Vomlel, 2005, Lemma 5.2).*
Let H be the EG of an equivalence class of DAGs over N such that $u_H \neq 0$. Then, for every $L \subseteq N$, exclusively *one of the following six cases occurs:*

(a) $L = core(H)$ *and* $u_H(L) = +1$,
(b) $L = K \cup pa_H(C)$ *for* $K \in \mathcal{K}(C)$, $C \in \mathcal{C}_{core}(H)$ *and* $u_H(L) = -1$,
(c) $L = S \cup pa_H(C)$ *for* $S \in \mathcal{S}(C)$, $C \in \mathcal{C}_{core}(H)$ *and* $u_H(L) = \nu_C(S) > 0$,
(d) $L = P$ *for* $P \in \mathcal{P}_{core}(H)$ *and* $u_H(L) = \tau(P) > 0$,
(e) $L = \emptyset$ *and* $u_H(L) = i(H) - 1 \geq 0$,
(f) *none of the above cases occurs and* $u_H(L) = 0$.

Lemma 7 implies that, given an EG H, the class of sets

$$\mathcal{K}_H \equiv \{K \cup pa_H(C); K \in \mathcal{K}(C), C \in \mathcal{C}_{core}(H)\}$$

and the class of sets

$$\mathcal{P}_{core}(H) \cup \mathcal{S}_H, \quad \text{where } \mathcal{S}_H \equiv \{S \cup pa_H(C); S \in \mathcal{S}(C), C \in \mathcal{C}_{core}(H)\},$$

can be determined on basis of u_H. Therefore, it follows from Lemma 7 that, given a non-zero standard imset, one can simply determine the core of the EG H, the number of its initial components, the collections of sets $\mathcal{K}_H$, and $\mathcal{P}_{core}(H) \cup \mathcal{S}_H$.

7 Reconstruction algorithm

Lemma 7 is a basis of a two-stage reconstruction algorithm for the EG from the standard imset; the proof of its correctness is quite long – see (Studený, Vomlel 2005).

The first stage of the algorithm is a *decomposition procedure*, whose output is an ordered sequence τ of subsets of the set of variables N. The procedure,

Table 4. Subroutine **Adapt** (u over $N \,|\, M$, w over M).

Input:	u	...	a standard imset over a non-empty set of variables N
Output:	M	...	a subset of N
	w	...	an adapted standard imset over M

1 find a maximal set $M \subseteq N$ with respect to inclusion with $u(M) \neq 0$;
2 $w :=$ the restriction of u to the class of subsets of M;
3 return M, w;

described in Table 6, consists of repeated application of two subroutines until one gets a zero imset. The first subroutine is an adaptation subroutine for (a standard imset) u which is applied if $u(N) = 0$ – see Table 4.

The basic idea of the second subroutine, which is described in Table 5, is to reduce the set of variables N. Thus, the original imset u over N is "decomposed" into an imset w over a proper subset $M \subset N$ and a certain set of nodes T with $M \cup T = N$. Note the set T chosen in line 4 of Table 5 plays crucial role in the reconstruction phase and one can prove that is a clique $T = K \cup pa_H(C)$, $K \in \mathcal{K}(C)$ of the closure graph for a component $C \in \mathcal{C}(H)$ that is a leaf-clique of a junction tree for cliques of $\bar{H}(C)$ – for details see (Studený and Vomlel, 2005, Section 7). The reduced imset w is obtained from the original imset u by subtracting a structural imset that corresponds to a CI statement $(M \setminus T) \perp\!\!\!\perp (T \setminus M) \,|\, (T \cap M)$ and by restricting to M – see lines 7-8 of Table 5.

Table 5. Subroutine **Reduce** (u over $N \,|\, T$, M, w over M).

Input:	u	...	an adapted standard imset over a non-empty set of variables N
Output:	T	...	a proper subset of the set of variables N
	M	...	a proper subset of N such that $M \cup T = N$ and $T \setminus M \neq \emptyset$
	w	...	a standard imset over M

1 $\mathcal{T} := \{L \subseteq N;\, u(L) < 0\}$;
2 $\mathcal{L} := \{L \subset N;\, u(L) > 0,\ L \neq \emptyset\}$;
3 $W := \bigcup \mathcal{L}$;
4 find $T \in \mathcal{T}$ such that $T \setminus W \neq \emptyset$ and $T \cap W \in \mathcal{L} \cup \{\emptyset\}$;
5 $R := T \setminus W$;
6 $M := N \setminus R$;
7 $\tilde{w} := u - \delta_N + \delta_M + \delta_T - \delta_{T \setminus R}$;
8 $w :=$ the restriction of $\tilde{w}$ to the class of subsets of M;
9 return T, M, w;

The decomposition procedure is illustrated by Example 23.

Table 6. The first stage **Decompose** (u over $N \,|\, \tau$ over N).

Input:	u ... a standard imset over a non-empty set of variables N
Output:	τ ... an ordered sequence of subsets of N

```
1    Y := N;
2    τ := empty list;
3    if u = 0 then
4        append Y as the last item in τ;
5        return τ;
6    if u ≠ 0 = u(Y) then
7        Adapt (u over Y | M, w over M);
8        append Y as the last item in τ;
9        go to 14;
10   if u(Y) ≠ 0 then
11       Reduce (u over Y | T, M, w over M);
12       append T as the last item in τ;
13       go to 14;
14   Y := M;
15   u := w;
16   go to 3;
17   exit;
```

Example 23. The ordered sequence of subsets that is the outcome of the first stage for the imset given in Table 3 is

$$\{a,b,c\},\{d,f,g\},\{a,b,d\},\{a,e\},\{b\},\{e,f\}\,. \tag{5}$$

Note that, in this case, only the subroutine **Reduce** was applied.

The basis of the dual procedure is the extension subroutine, described in Table 7. It constructs the EG H over N on basis of its induced subgraph G for $M \subset N$ and a set $T \subseteq N$ with $M \cup T = N$. Of course, the set is assumed to have above-mentioned special form $T = K \cup pa_H(C)$. The crucial step to fully reconstruct H on basis of G and T is to decide which of two following cases occurs: either $T \cap M = pa_H(C)$ or $T \cap M = Z \cup pa_H(C)$, where $Z \in \mathcal{S}(C)$. The condition in line 6 of Table 7 is a necessary and sufficient condition for the second case.

Now, the second stage of the algorithm, the *composition procedure*, which consists of repeated application of the subroutine **Extend**. It is described in Table 8. Its input is the ordered sequence τ of sets obtained from the decomposition procedure. However, the sequence τ is processed in the reverse order. The procedure is illustrated by Example 24.

Example 24. The output of the second stage of the algorithm applied to the ordered sequence (5) of subsets of N is the EG in Figure 5.

Table 7. Subroutine **Extend** (G over M, $T \,|\, H$ over $M \cup T$).

Input:	G	... a chain graph over a non-empty set of variables M
	T	... a set of variables with $T \setminus M \neq \emptyset$
Output:	H	... a chain graph over $M \cup T$

```
1   L := T ∩ M;
2   R := T \ M;
3   Z := ∅;
4   if L ≠ ∅ then
5       choose a terminal component X in G_L;
6   if X is a complete component in G_L and pa_G(X) = L \ X
7       then put Z := X;
8   determine the edges in H as follows:
9       H_M := G;
10      ∀ x ∈ L \ Z, ∀ z ∈ R include x → z in H;
11      ∀ y ∈ R ∪ Z, z ∈ R, y ≠ z include y — z in H;
12  return H;
```

Table 8. The second stage **Compose** (τ over $N \,|\, H$ over N).

Input:	τ	... an ordered sequence $T_1, \ldots, T_n$, $n \geq 1$ of subsets of N
Output:	H	... a chain graph over M

```
1   M := T_n;
2   H := the complete undirected graph over M;
3   G := H
4   for j = n − 1, ..., 1 do
5       Extend (G over M, T_j | H over M ∪ T_j)
6       M := M ∪ T_j;
7       G := H;
8   return H;
```

8 Construction of a hierarchical junction tree

The sequence of sets that is the outcome of the first stage (Table 6) can also be used to construct a hierarchical junction tree similar to those introduced in Puch et al. (2004).

Each set in the sequence defines a node of the hierarchical junction tree whose entering edges could be labeled by sets Z or L obtained during the second stage of the reconstruction algorithm (see Table 8). More specifically, each node T may or may not be ascribed an entering edge and the edge can be labeled either by a separator Z (if $Z \neq \emptyset$) or by a parent set L (if $Z = \emptyset$ and $L \equiv L \setminus Z \neq \emptyset$). These units can be used then to compose the whole

hierarchical junction tree. Due to the lack of space we omit details of the construction and give two examples instead.

Example 25. The nodes of a hierarchical junction tree constructed from the sequence (5) are given in Figure 6, including their attributed separators and parent sets. The resulting hierarchical junction tree is given in Figure 7.

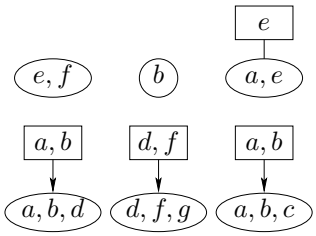

Fig. 6. Units of a hierarchical junction tree.

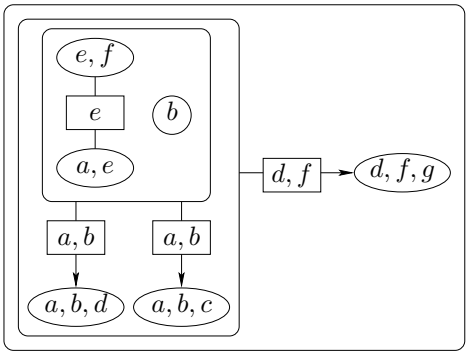

Fig. 7. A hierarchical junction tree.

Example 26. Figure 8 gives another example of an EG H. The first stage of the reconstruction algorithm applied to the standard imset u_H ends with the sequence of sets

$$\{a,b,c,d\},\ \{a,b,d,e\},\ \{a\},\ \{b\}\,.$$

The nodes of the respective hierarchical junction tree are given in Figure 9, including their attributed separators and parent sets. The resulting hierarchical junction tree is given in Figure 10. In that picture, the parent set $pa_H(\{c,d,e\}) = \{a,b\}$ is attributed to just one node of the hierarchical junction tree. One can perhaps draw a picture in which every parent set is ascribed to every node of the respective component of the hierarchical junction tree.

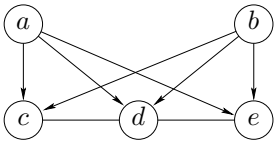

Fig. 8. An EG.

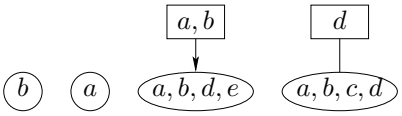

Fig. 9. Units of a hierarchical junction tree.

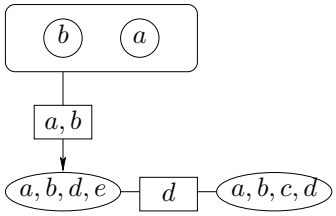

Fig. 10. A hierarchical junction tree.

Conclusions

The presented procedures for the transition between graphical and algebraic representatives of a CI model generated by DAG. These can be the first step on the way towards a fully algebraic method for learning structure of Bayesian networks. We hope that the procedures can be utilized to find a characterization of the inclusion neighborhood of a given DAG in terms of the standard imset. This will be a topic of a future research. We also plan to study the polytope generated by standard imsets over N hoping that linear programming maximization methods can be applied in learning Bayesian networks.

Acknowledgments

This research has been supported by the grant GAČR nr. 201/04/0393.

Bibliography

Andersson SA, Madigan D, and Perlman MD. A characterization of Markov equivalence classes for acyclic digraphs. *Annals of Statistics*, 25:pp. 505–541 (1997a).

Andersson SA, Madigan D, and Perlman MD. On the Markov equivalence of chain graphs, undirected graphs and acyclic digraphs. *Scandinavian Journal of Statistics*, 24:pp. 81–102 (1997b).

Chickering DM. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:pp. 507–554 (2002).

Jensen FV. *Bayesian Networks and Decision Graphs*. Springer Verlag (2001).

Kočka T and Castelo R. Improved learning of Bayesian networks. In J Breese and D Koller, eds., *Uncertainty in Artificial Intelligence 17*, pp. 269–276. Morgan Kaufmann (2001).

Lauritzen SL. *Graphical Models*. Clarendon Press (1996).

Lauritzen SL and Wermuth N. Graphical models for associations between variables some of which are qulitative and some quantitative. *Annals of Statistics*, 17:pp. 31–57 (1989).

Pearl J. *Probabilistic Reasoning in Intelligent Systems - Networks of Plausible Inference*. Morgan Kaufmann (1988).

Puch RO, Smith JQ, and Bielza C. Hierarchical junction trees: conditional independence preservation and forecasting in dynamic Bayesian networks with heterogeneous evolution. In JA Gámez, S Moral, and A Salmerón, eds., *Advances in Bayesian Networks*, pp. 57–75. Springer-Verlag (2004).

R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria (2004). ISBN 3-900051-00-3.

Studený M. Characterization of essential graphs by means of the operation of legal merging of components. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 12:pp. 43–62 (2004).

Studený M. Characterization of inclusion neighbourhood in terms of the essential graph. *International Journal of Approximate Reasoning*, 38:pp. 283–309 (2005a).

Studený M. *On Probabilistic Conditional Independence Structures*. Springer-Verlag (2005b).

Studený M, Roverato A, and Štěpánová Š. Two operations of merging components in a chain graph. *Scandinavian Journal of Statistics* (2005). Submitted.

Studený M and Vomlel J. A reconstruction algorithm for the essential graph. Tech. rep., ÚTIA AV ČR (2005). `http://staff.utia.cz/vomlel/sv.ps`.

Verma T and Pearl J. Equivalence and synthesis of causal models. In PP Bonissone, M Henrion, LN Kanal, and JF Lemmer, eds., *Uncertainty in Artificial Intelligence 6*, pp. 220–227. Elsevier (1991).